Toward Implementation of Artificial Neural Networks That "Really Work"

Mauricio A. Leon, MD and James Keller, Ph.D.

From Medical Informatics-ITS and

The Department of Computer Engineering and Computer Science

University of Missouri-Columbia

Columbia, Missouri 65211

Artificial neural networks are established analytical methods in bio-medical research. They have repeatedly outperformed traditional tools for pattern recognition and clinical outcome prediction while assuring continued adaptation and learning. However, successful experimental neural networks systems seldom reach a production state. That is, they are not incorporated into clinical information systems. It could be speculated that neural networks simply must undergo a lengthy acceptance process before they become part of the day to day operations of health care systems. However, our experience trying to incorporate experimental neural networks into information systems lead us to believe that there are technical and operational barriers that greatly difficult neural network implementation. A solution for these problems may be the delineation of policies and procedures for neural network implementation and the development a new class of neural network client/server applications that fit the needs of current clinical information systems.

INTRODUCTION

Extensive research had confirmed the utility of artificial neural networks for the solution of clinical diagnostics and prognostic problems[1-11]. However, there is practically no indication that these technologies are being incorporated into clinical information systems or embedded into stand-alone instruments. Our efforts to incorporate some successful experimental neural networks into existing clinical information systems[12-16] has allowed us to understand a bit better the difficulties of neural network deployment in clinical information systems. A working definition for a deployed neural network system could be *a neural network system that is integrated to a clinical information system and delivers information to caregivers and administrators, while preserving its ability to learn and adapt.* While resistance to new technology can be argued as the reason for lack of neural network deployment, we suggest that there are several outstanding technical and operational problems that prevent application of neural networks into clinical information systems. The scope of this paper is to describe some of these problems and to present the solutions that we have considered and developed.

BACKGROUND

Common goals in medical care are to diagnose or predict conditions that afflict patients (i.e. whether a patient will develop a nosocomial infection) or that affect the organization (i.e. whether a patient will maintain his scheduled appointment). Classic statistical methods and neural networks can be used to reach these goals.

Statistical methods have been used extensively in the medical domain. Some models developed using statistics methods can be "packed" into rules, score systems, or equations that health care providers can apply in their daily activities. Examples of such models include cancer survival tables, trauma severity scores, newborn status scores, and many more. When available, these models are useful and easy to use. Unfortunately, they are static, their validity has to be verified regularly, they may not apply to populations other than the ones studied, and the models themselves must be very simple in order to gain acceptance by the health care community.

Conversely, neural networks are relatively new modeling techniques. They have been applied in medical research mostly to develop diagnostic and prognostic models for problems that could not be handled easily with more traditional options. However, neural networks are also used to solve problems for which traditional modeling has been employed previously. Almost always, neural network models had outperformed or at least matched their statistical counterparts. In addition to higher accuracy, a common claim from neural network researchers is that these applications can maintain or even increase their diagnostic or prognostic performance as more experience is accumulated. Neural networks can adapt to the population that they receive data from. Unfortunately, neural networks

1091—8280/97/$5.00

cannot be simplified into rules or score systems, and they have to be deployed only as computer-based solutions.

Mostly, methodologies for the utilization of statistical and neural networks tools are identical: Data has to be collected and pre-processed; the analytical and processing options have to be defined; data has to be entered and processed by the system; and finally the system's performance has to be evaluated.

BARRIERS FOR NEURAL NETWORK DEPLOYMENT

When a neural network has demonstrated its usefulness in an experimental setting, it is desirable to *deploy* it. Deployment means that the neural network becomes a permanent decision support resource for administrative and healthcare personnel. We have identified barriers for deployment that can be grouped into technical and operational categories.

Technical barriers

Technical barriers originate in part from the inability of commercially available neural network packages to support functionality that must be available in a deployable neural network system. So far, we have identified the need to provide functionality in the following areas:

Data interface. While most commercial neural network packages offer mechanisms to import data into the application, none seem to offer the kind of interfaces that are common among clinical information systems. The data interface must link the neural network system to any data source inside a clinical information network. The data interface must be able to "speak" all industry accepted medical data communication protocols and encoding systems. Data interface functions should enable the neural network system to identify new data and to fetch them from their respective databases. Furthermore, a neural network system must use its data interface to notify other systems or even individual care givers of its diagnostic or predictive findings.

Data conditioning. Data conditioning functionality of deployable neural networks differ from that of commercial packages in the timing and manner in which these functions can be applied. In most research where neural networks are applied, all data is available before neural network training is even started. In these cases, train, test and validation data are all processed jointly, and the resulting coefficients remain constant for the life of the experiment. The challenge for a continuously adapting neural network is that the actual range and distribution of input data may not be known. However, data still have to be pre-processed somehow to be feed into the network. In a deployable neural network system, data conditioning functions must be applied dynamically, perhaps every time that new data is available.

Database management. In addition to a robust data interface, a deployed neural network system requires strong database management functionality for two main reasons: to link to external databases and to maintain an internal data model. In the fist case, the system needs to provide support for query development and implementation. For example, a user developing a new neural network should be able to inspect remote databases and to graphically design the queries that retrieve the information from them. In the second case, the database management functions are used store all past and present data relevant to the system. These data include model definition information, model state, user bases, copies of training and testing data, error logs, access logs, etc.

System performance evaluation. In most neural network applications performance is measured determining the accuracy of the network on a test data set for which the correct outcome is known. This popular approach relies on the assumption that patterns to be learned are present in both train and data sets, and consequently test samples should not contribute significantly to model enhancement. While this practice is already questionable in "traditional" neural network applications, it becomes more difficult to apply in deployed networks that must respond to input analysis demands on-line. In these systems the idea of static training and data sets may have to be replaced by a more dynamic approach. In this approach, all data first belongs to a *challenge* set. This set contains data that is supplied to the system without knowledge of the actual outcome. As time passes, more information is added to the clinical information system, and the actual outcome of the date in the challenge set becomes available. At this moment, the challenge set becomes the test set, because performance can be measured. Finally the patterns in the test set that were misclassified can be used to retrain the network, thus becoming part of the train set. This dynamic approach the cumulative performance of the network can be measured continuously.

Scalability and long term system behavior. If a neural network is allowed to continue training, there is the underlying assumption that the system's performance may improve by learning new patterns or classifying better those patterns previously learned. However, there is the risk that a fixed neural

network architecture could reach a point where further improvement is not possible. In this situation, the neural network may be unable to learn new associations, or may do so only by dropping some of the knowledge previously acquired. To avoid this situation the neural network must be provided with a mechanism to enhance its own structure. A solution is to deploy a neural network in a system capable of training simultaneously several predictive models. These models are themselves neural networks that, in their simplest form, differ in the number of nodes and layers. More complicated alternative models may include neural networks that can split large nets into small "specialized" ones; neural nets that try different paradigms until a more successful solution is achieved; or perhaps a combination of all previous alternatives.

Error recovery. A deployed neural network system must provide mechanisms to identify and solve errors due to learning, application software, or machine failures. The most complete method for error recovery requires the creation of a transaction database that keeps track of all committed transactions. However, this approach may be prohibitively expensive because even the smallest change of a single weight may be considered a committed transaction. A scaled down version of this approach could be a database that just keeps track of the valid states of the system. A valid state may be defined as a snapshot of all system variables and data, at a time when the system is not in error, that allows the recreation of the system. Similar to database systems, the reach and sophistication of the error recovery mechanisms should be tailored to the importance of the data handled.

Knowledge recovery. Knowledge recovery is important because neural networks don not execute rigid guidelines. Neural networks evolve, and therefore they may respond differently to the same stimuli if it is presented at different evolutionary stages. For legal and evaluation purposes, it may be necessary to replicate exactly the behavior of a network at any point in its developmental process. Error recovery functionality in a deployed neural network should enable a set of knowledge recovery functions. Knowledge recovery is defined here as the ability of the system to recall an earlier knowledge status. For example, a neural network that has been learning for several years should be able to recall the knowledge that it accumulated precisely at the end of the first year of training. As in the case of error recovery, the sophistication of the knowledge recovery features must be tied to the character of the data.

Event logging and security. As a tool for decision support, a deployed neural network system may be required to maintain an accurate record of all user events. Broadly, neural network user events may be classified as administrative and consults. Administrative events include all user actions that affect the neural network definition, implementation, and management. Consults are user events characterized by the delivery of neural network *output* data to users. Consults may be initiated by individual users or may occur automatically if the neural network system has been instructed to broadcast its findings.

Client/Server architecture. A deployable neural network system probably benefits from a client server design for security and performance reasons. For example it is likely that the computational demand of continuously running diagnostic or predictive neural networks may exceed individual user computing capabilities. Further, it is conceivable that deployed neural networks could run more cost/efficiently in highly parallel machines that may have to be supported by specialized departments.

Operational barriers

Operational barriers originate from the very *learning* nature of neural networks. Deployed neural networks become evolving systems that require *continuous* guidance and supervision. Thus, these systems necessitate *supervisors* who are individuals that must spend time regularly verifying the performance and responding to the learning demands of the system. Operational barriers rise if the supervisory role is neglected. Some of the responsibilities of a supervisor comprise:

Definition/Termination of neural network projects. A deployable neural network system must offer supervisor users the option of defining new neural network projects. These projects can be of at least two broad types: implementation of preexisting neural networks, and definition of new ones. In both cases, a supervisor must define the behavior and evolution parameters of the neural network project.

Validation of data. Even if the neural network system is able to obtain data from the clinical information system. The decision of whether that data must be included in the test and training sets may have to be taking by a supervisor. In this scenario, the system should notify the supervisor user of the availability of new data and the need for confirmation.

Selection of models. A deployable neural network system may accomplish its continuous learning and adaptation goal by permanently trying new models in

the "background". If one of these models outperform the currently accepted model, then the user supervisor may have to be notified and asked to authorize the implementation of the newer model. This is particularly important if the old and new models differ considerably.

Definition of output evaluation criteria. The supervisor must define the criteria that the network will use to judge its performance.

Error management. The supervisor must instruct the neural network system to indicate when a learning error is encountered or suspected. An erroneous pattern may be present when two patterns share the same input data but lead to different outcomes, or when a new pattern can not be incorporated into the system after an extensive training has been attempted. The supervisor will have to inspect the error candidates offered by the network and will have to make a decision about the validity of the data. Error recovery mechanisms can be applied in this situation.

A MODEL FOR DEPLOYABLE NEURAL NETS

A prototype of a deployable neural network system was developed to investigate the problems associated with the integration of neural network to clinical information systems. This prototype was written in C language and consisted of client and server components. The client was provided with just enough functionality to submit neural network definition information to the server side. The server incorporated essential components of the data interface, data conditioning, database management, event logging, and output data evaluation features described before. The server also contained a robust neural network manager that was able define, launch, and control parallel multilayer backpropagation networks executing in local and remote machines.

METHODS

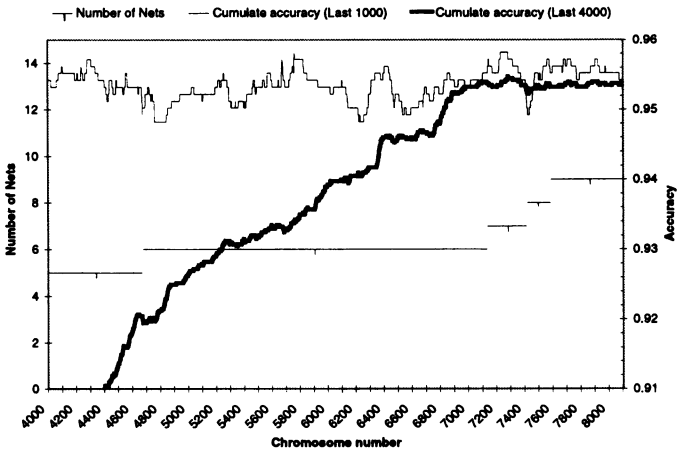
The system was tested with a chromosome recognition problem, which is an ideal domain for continuous learning and adaptation. The objective of the test was to determine if the system could achieve the recognition performance of "traditional" neural networks while maintaining its on-line learning ability. The system was instructed to: (1) obtain data from a database containing the Copenhagen chromosome data set (in this set, each chromosome is represented by a 31 element vector: the first 30 elements corresponded to input features, and the last element is the chromosome class) (2) Create a single, small backpropagation neural network (30 input nodes, 5 hidden nodes, and 24 output nodes) (3) Create new networks as needed. (4) Report its performance as the cumulated accuracy in the last 1000 and 4000 analyzed chromosomes, respectively. Additionally, a program was written to populate the database in a progressive manner: one chromosome was stored after the other, and class information was update long after input data has been instantiated (this was done to simulate the normal flow of data in a clinical environment)

The neural network system started by initializing the single backpropagation neural net, and then it queried the database to determine data availability. When input data of the first record in the database became available, the system downloaded it. The partial record was instantiated into a local database, scaled, and supplied to the backpropagation network. The response of the network was stored and the system resumed searching the database for the class information of the existing record or for more records if available. When class information for the first record was available, the system fetched it and compared to the output that was generated before. If the output was correct, the system resumed the search for new data. If the result was incorrect, (and it was!) then the record was tagged as training data, and the network started training until all training records were recognized correctly. When training was finished the system resumed its search for data. When data was available the entire process was repeated. After a while, the small neural network was unable to learn to classify more chromosomes correctly. The system notified the supervisor, and asked whether to continue trying with the same network or whether to create an additional, parallel network that could specialize in those chromosomes that were not learnable by the existing network. In the current test, the system was always authorized to create additional networks. The process stopped when the system processed the 8106 chromosomes that were asynchronically instantiated into the database.

RESULTS

The figure shows the cumulated performance of the system during processing of the last 4000 chromosomes. When the last chromosome was processed the cumulated accuracy of the previous 4000 analyzes was 95.3%. This number is greater than the 94.1% level obtained using a using single neural network in a conventional training/testing approach[12]. Interestingly the performance of the cumulated previous 1000 analyzes was even higher.

The total number of networks required at the end of the test were 9, and the sum of hidden neurons across networks was 45. This number matches exactly the number of hidden neurons in the single neural network model.



DISCUSION

The results indicate that the performance of a deployable neural network may be unaffected or increased when compared to the traditional neural network training methodologies. These are good news when considering that the deployable system maintains its growth and adaptation potential. Incorporation of neural networks into clinical information systems is an uncharted territory. The suggestions mentioned in this working paper are partly a result from the problems that we have encountered trying to make our neural network applications "really work", and partly a preemptive response to the problems that we foresee for larger more sophisticated prediction/diagnostic neural nets. Regardless of the final characteristics, definition, and name of the entity here called "deployable neural network", we anticipate that clinical information systems will embrace these systems when they are ready to deliver.

Acknowledgments:

This work was supported, in part, by a training grant #LM-07089, from the National Library of Medicine, National Institutes of Health, Bethesda, MD

References:

1. Becker RL JR. Applications of neural networks in histopathology. Pathologica, 1995: 87(3):246-54.
2. Burke HB. Artificial neural networks for cancer research: outcome prediction. Seminars in Surgical Oncology, 1994: 10(1):73-9.
3. Florio T. Einfeld S. Levy F. Neural networks and psychiatry: candidate applications in clinical. decision making. Australian & New Zealand Journal of Psychiatry, 1994:28(4):651-66.
4. Forsstrom JJ. Dalton KJ. Artificial neural networks for decision support in clinical medicine. Annals of Medicine, 1995:27(5):509-17.
5. Galletly CA. Clark CR. McFarlane AC. Artificial neural networks: a prospective tool for the analysis of psychiatric disorders. Journal of Psychiatry & Neuroscience, 1996:21(4):239-47.
6. Itchhaporia D. Snow PB. Almassy RJ. Oetgen WJ. Artificial neural networks: current status in cardiovascular medicine. Journal of the American College of Cardiology, 1996: 28(2):515-21.
7. Montague G. Morris J. Neural-network contributions in biotechnology. Trends In Biotechnology, 1994:12(8):312-24.
8. Mylrea KC. Orr JA. Westenskow DR. Integration of monitoring for intelligent alarms in anesthesia: neural networks--can they help?. Journal of Clinical Monitoring, 1993: 9(1):31-7.
9. Reggia JA. Neural computation in medicine. Artificial Intelligence in Medicine, 1993:5(2):143-57.
10. Winkel P. Artificial intelligence within the chemical laboratory. Annales de Biologie Clinique, 1994:2(4):277-82.
11. Räsänen J, León MA, Neural networks in Critical Care. In: J.L. Vincent (ed) 1995 Yearbook of intensive care and emergency medicine. Springer-Verlag. Berlin.
12. León MA. Gader P. Keller J. Multiple neural network response variability as a predictor of neural network accuracy for chromosome recognition. Biomedical Sciences Instrumentation, 1996:32:31-7.
13. León MA. Rasanen J. Neural network-based detection of esophageal intubation in anesthetized patients. Journal of Clinical Monitoring, 1996:12(2):165-9.
14. León MA. Rasanen J. Mangar D. Neural network-based detection of esophageal intubation. Anesthesia & Analgesia, 1994 78(3):548-53.
15. León MA. Use of artificial neural-networks for monitoring inspiratory work of breathing during pressure support ventilation. Anesthesia & Analgesia, 1993:76(S):S218
16. León MA, Räsänen J. Neural network assessment of respiratory system mechanics. Intensive Care Medicine, 1992:18(S2):256.